Elektrisk Mätteknik, LTH

Datorbaserade mätsystem

5. Mätdatabehandling

06-08-30

NATIONAL INSTRUMENTS™

Application Note 041

The Fundamentals of FFT-Based Signal Analysis and Measurement

Michael Cerna and Audrey F. Harvey

Introduction

The Fast Fourier Transform (FFT) and the power spectrum are powerful tools for analyzing and measuring signals from plug-in data acquisition (DAQ) devices. For example, you can effectively acquire time-domain signals, measure the frequency content, and convert the results to real-world units and displays as shown on traditional benchtop spectrum and network analyzers. By using plug-in DAQ devices, you can build a lower cost measurement system and avoid the communication overhead of working with a stand-alone instrument. Plus, you have the flexibility of configuring your measurement processing to meet your needs.

To perform FFT-based measurement, however, you must understand the fundamental issues and computations involved. This application note serves the following purposes.

- Describes some of the basic signal analysis computations,
- Discusses antialiasing and acquisition front ends for FFT-based signal analysis,
- Explains how to use windows correctly,
- Explains some computations performed on the spectrum, and
- Shows you how to use FFT-based functions for network measurement.

The basic functions for FFT-based signal analysis are the FFT, the Power Spectrum, and the Cross Power Spectrum. Using these functions as building blocks, you can create additional measurement functions such as frequency response, impulse response, coherence, amplitude spectrum, and phase spectrum.

FFTs and the Power Spectrum are useful for measuring the frequency content of stationary or transient signals. FFTs produce the average frequency content of a signal over the entire time that the signal was acquired. For this reason, you should use FFTs for stationary signal analysis or in cases where you need only the average energy at each frequency line. To measure frequency information that is changing over time, use joint time-frequency functions such as the Gabor Spectrogram.

This application note also describes other issues critical to FFT-based measurement, such as the characteristics of the signal acquisition front end, the necessity of using windows, the effect of using windows on the measurement, and measuring noise versus discrete frequency components.

National Instruments™, ni.com™, and LabWindows/CVI™ are trademarks of National Instruments Corporation. Product and company names mentioned herein are trademarks or trade names of their respective companies.

 © Copyright 2000 National Instruments Corporation. All rights reserved.

Basic Signal Analysis Computations

The basic computations for analyzing signals include converting from a two-sided power spectrum to a single-sided power spectrum, adjusting frequency resolution and graphing the spectrum, using the FFT, and converting power and amplitude into logarithmic units.

The power spectrum returns an array that contains the two-sided power spectrum of a time-domain signal. The array values are proportional to the amplitude squared of each frequency component making up the time-domain signal. A plot of the two-sided power spectrum shows negative and positive frequency components at a height

$$\frac{A_k^2}{4}$$

where A_k is the peak amplitude of the sinusoidal component at frequency *k*. The DC component has a height of A_0^2 where A_0 is the amplitude of the DC component in the signal.

Figure 1 shows the power spectrum result from a time-domain signal that consists of a 3 Vrms sine wave at 128 Hz, a 3 Vrms sine wave at 256 Hz, and a DC component of 2 VDC. A 3 Vrms sine wave has a peak voltage of $3.0 \bullet \sqrt{2}$ or about 4.2426 V. The power spectrum is computed from the basic FFT function. Refer to the Computations Using the FFT section later in this application note for an example this formula.

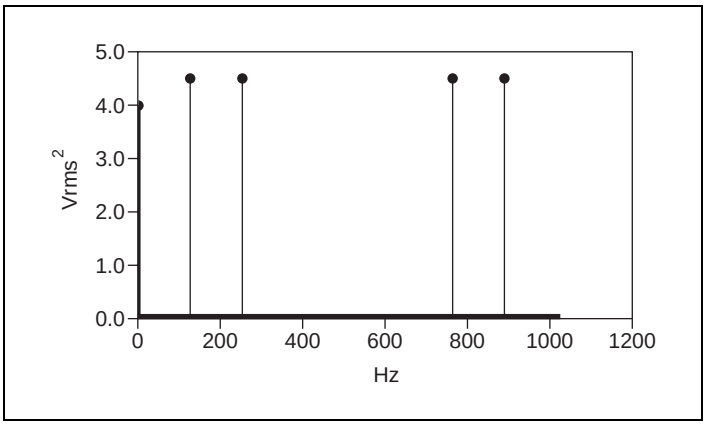


Figure 1. Two-Sided Power Spectrum of Signal

Converting from a Two-Sided Power Spectrum to a Single-Sided Power Spectrum

Most real-world frequency analysis instruments display only the positive half of the frequency spectrum because the spectrum of a real-world signal is symmetrical around DC. Thus, the negative frequency information is redundant. The two-sided results from the analysis functions include the positive half of the spectrum followed by the negative half of the spectrum, as shown in Figure 1.

In a two-sided spectrum, half the energy is displayed at the positive frequency, and half the energy is displayed at the negative frequency. Therefore, to convert from a two-sided spectrum to a single-sided spectrum, discard the second half of the array and multiply every point except for DC by two.

$$G_{AA}(i) = S_{AA}(i),\ i = 0\ (DC)$$

$$G_{AA}(i) = 2 \bullet S_{AA}(i),\ i = 1 \text{ to } \frac{N}{2} - 1$$

where S_{AA}(i) is the two-sided power spectrum, G_{AA}(i) is the single-sided power spectrum, and N is the length of the two-sided power spectrum. The remainder of the two-sided power spectrum S_{AA}

$$\left(\frac{N}{2} \text{ through } N-1\right)$$

is discarded.

The non-DC values in the single-sided spectrum are then at a height of

$$\frac{A_k^2}{2}$$

This is equivalent to

$$\left(\frac{A_k}{\sqrt{2}}\right)^2$$

where

$$\frac{A_k}{\sqrt{2}}$$

is the root mean square (rms) amplitude of the sinusoidal component at frequency k. Thus, the units of a power spectrum are often referred to as quantity squared rms, where quantity is the unit of the time-domain signal. For example, the single-sided power spectrum of a voltage waveform is in volts rms squared.

Figure 2 shows the single-sided spectrum of the signal whose two-sided spectrum Figure 1 shows.

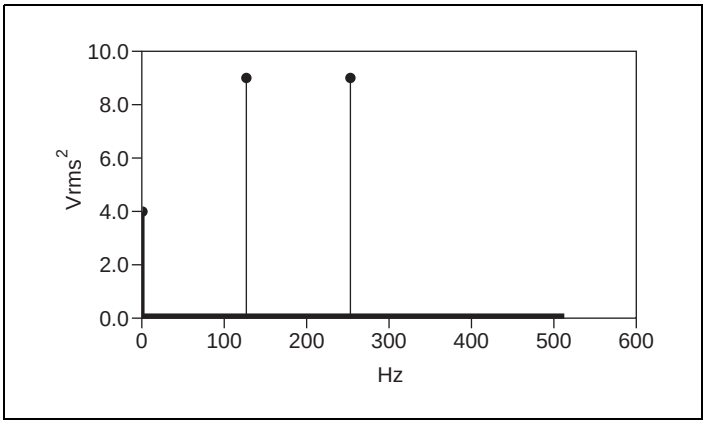


Figure 2. Single-Sided Power Spectrum of Signal in Figure 1

As you can see, the level of the non-DC frequency components are doubled compared to those in Figure 1. In addition, the spectrum stops at half the frequency of that in Figure 1.

Adjusting Frequency Resolution and Graphing the Spectrum

Figures 1 and 2 show power versus frequency for a time-domain signal. The frequency range and resolution on the x-axis of a spectrum plot depend on the sampling rate and the number of points acquired. The number of frequency points or lines in Figure 2 equals

$$\frac{N}{2}$$

where N is the number of points in the acquired time-domain signal. The first frequency line is at 0 Hz, that is, DC. The last frequency line is at

$$\frac{F_s}{2} - \frac{F_s}{N}$$

where F_s is the frequency at which the acquired time-domain signal was sampled. The frequency lines occur at Δf intervals where

$$\Delta f = \frac{F_s}{N}$$

Frequency lines also can be referred to as frequency bins or FFT bins because you can think of an FFT as a set of parallel filters of bandwidth Δf centered at each frequency increment from

$$\text{DC to } \frac{F_s}{2} - \frac{F_s}{N}$$

Alternatively you can compute Δf as

$$\Delta f = \frac{1}{N \bullet \Delta t}$$

where Δt is the sampling period. Thus $N \bullet \Delta t$ is the length of the time record that contains the acquired time-domain signal. The signal in Figures 1 and 2 contains 1,024 points sampled at 1.024 kHz to yield Δf = 1 Hz and a frequency range from DC to 511 Hz.

The computations for the frequency axis demonstrate that the sampling frequency determines the frequency range or bandwidth of the spectrum and that for a given sampling frequency, the number of points acquired in the time-domain signal record determine the resolution frequency. To increase the frequency resolution for a given frequency range, increase the number of points acquired at the same sampling frequency. For example, acquiring 2,048 points at 1.024 kHz would have yielded Δf = 0.5 Hz with frequency range 0 to 511.5 Hz. Alternatively, if the sampling rate had been 10.24 kHz with 1,024 points, Δf would have been 10 Hz with frequency range from 0 to 5.11 kHz.

Computations Using the FFT

The power spectrum shows power as the mean squared amplitude at each frequency line but includes no phase information. Because the power spectrum loses phase information, you may want to use the FFT to view both the frequency and the phase information of a signal.

The phase information the FFT yields is the phase relative to the start of the time-domain signal. For this reason, you must trigger from the same point in the signal to obtain consistent phase readings. A sine wave shows a phase of –90° at the sine wave frequency. A cosine shows a 0° phase. In many cases, your concern is the relative phases between components, or the phase difference between two signals acquired simultaneously. You can view the phase difference between two signals by using some of the advanced FFT functions. Refer to the *FFT-Based Network Measurement* section of this application note for descriptions of these functions.

The FFT returns a two-sided spectrum in complex form (real and imaginary parts), which you must scale and convert to polar form to obtain magnitude and phase. The frequency axis is identical to that of the two-sided power spectrum. The amplitude of the FFT is related to the number of points in the time-domain signal. Use the following equation to compute the amplitude and phase versus frequency from the FFT.

$$\text{Amplitude spectrum in quantity peak} = \frac{\text{Magnitude [FFT(A)]}}{N} = \frac{\sqrt{[\text{real}[\text{FFT}(\text{A})]]^2 + [\text{imag}[\text{FFT}(\text{A})]]^2}}{N}$$

$$\text{Phase spectrum in radians} = \text{Phase [FFT(A)]} = \text{arctangent}\left(\frac{\text{imag}[\text{FFT}(\text{A})]}{\text{real}[\text{FFT}(\text{A})]}\right)$$

where the arctangent function here returns values of phase between $-\pi$ and $+\pi$, a full range of 2π radians.

Using the rectangular to polar conversion function to convert the complex array

$$\frac{\text{FFT(A)}}{N}$$

to its magnitude (r) and phase (ø) is equivalent to using the preceding formulas.

The two-sided amplitude spectrum actually shows half the peak amplitude at the positive and negative frequencies. To convert to the single-sided form, multiply each frequency other than DC by two, and discard the second half of the array. The units of the single-sided amplitude spectrum are then in quantity peak and give the peak amplitude of each sinusoidal component making up the time-domain signal. For the single-sided phase spectrum, discard the second half of the array.

To view the amplitude spectrum in volts (or another quantity) rms, divide the non-DC components by the square root of two after converting the spectrum to the single-sided form. Because the non-DC components were multiplied by two to convert from two-sided to single-sided form, you can calculate the rms amplitude spectrum directly from the two-sided amplitude spectrum by multiplying the non-DC components by the square root of two and discarding the second half of the array. The following equations show the entire computation from a two-sided FFT to a single-sided amplitude spectrum.

$$\text{Amplitude spectrum in volts rms} = \sqrt{2} \bullet \frac{\text{Magnitude[FFT(A)]}}{N} \quad \text{for } i = 1 \text{ to } \frac{N}{2} - 1$$

$$= \frac{\text{Magnitude[FFT(A)]}}{N} \quad \text{for } i = 0 \text{ (DC)}$$

where *i* is the frequency line number (array index) of the FFT of A.

The magnitude in volts rms gives the rms voltage of each sinusoidal component of the time-domain signal.

To view the phase spectrum in degrees, use the following equation.

$$\text{Phase spectrum in degrees} = \frac{180}{\pi} \bullet \text{Phase FFT(A)}$$

The amplitude spectrum is closely related to the power spectrum. You can compute the single-sided power spectrum by squaring the single-sided rms amplitude spectrum. Conversely, you can compute the amplitude spectrum by taking the square root of the power spectrum. The two-sided power spectrum is actually computed from the FFT as follows.

$$\text{Power spectrum } S_{AA}(f) = \frac{FFT(A) \bullet FFT^*(A)}{N}$$

where FFT*(A) denotes the complex conjugate of FFT(A). To form the complex conjugate, the imaginary part of FFT(A) is negated.

When using the FFT in LabVIEW, be aware that the speed of the power spectrum and the FFT computation depend on the number of points acquired. If *N* is a power of two, LabVIEW uses the efficient FFT algorithm. Otherwise, LabVIEW actually uses the discrete Fourier transform (DFT), which takes considerably longer. LabWindows/CVI requires that *N* be a factor of two and thus always uses the FFT. Typical benchtop instruments use FFTs of 1,024 and 2,048 points.

So far, you have looked at display units of volts peak, volts rms, and volts rms squared, which is equivalent to mean-square volts. In some spectrum displays, the rms qualifier is dropped for Vrms, in which case V implies Vrms, and V^2 implies $Vrms^2$, or mean-square volts.

Converting to Logarithmic Units

Most often, amplitude or power spectra are shown in the logarithmic unit decibels (dB). Using this unit of measure, it is easy to view wide dynamic ranges; that is, it is easy to see small signal components in the presence of large ones. The decibel is a unit of ratio and is computed as follows.

$$dB = 10\log_{10} P/P_r$$

where P is the measured power and P_r is the reference power.

Use the following equation to compute the ratio in decibels from amplitude values.

$$dB = 20\log_{10} A/A_r$$

where A is the measured amplitude and A_r is the reference amplitude.

When using amplitude or power as the amplitude-squared of the same signal, the resulting decibel level is exactly the same. Multiplying the decibel ratio by two is equivalent to having a squared ratio. Therefore, you obtain the same decibel level and display regardless of whether you use the amplitude or power spectrum.

As shown in the preceding equations for power and amplitude, you must supply a reference for a measure in decibels. This reference then corresponds to the 0 dB level. Several conventions are used. A common convention is to use the reference 1 Vrms for amplitude or 1 Vrms squared for power, yielding a unit in dBV or dBVrms. In this case, 1 Vrms corresponds to 0 dB. Another common form of dB is dBm, which corresponds to a reference of 1 mW into a load of 50 Ω for radio frequencies where 0 dB is 0.22 Vrms, or 600 Ω for audio frequencies where 0 dB is 0.78 Vrms.

Antialiasing and Acquisition Front Ends for FFT-Based Signal Analysis

FFT-based measurement requires digitization of a continuous signal. According to the Nyquist criterion, the sampling frequency, F_s, must be at least twice the maximum frequency component in the signal. If this criterion is violated, a phenomenon known as aliasing occurs. Figure 3 shows an adequately sampled signal and an undersampled signal. In the undersampled case, the result is an aliased signal that appears to be at a lower frequency than the actual signal.

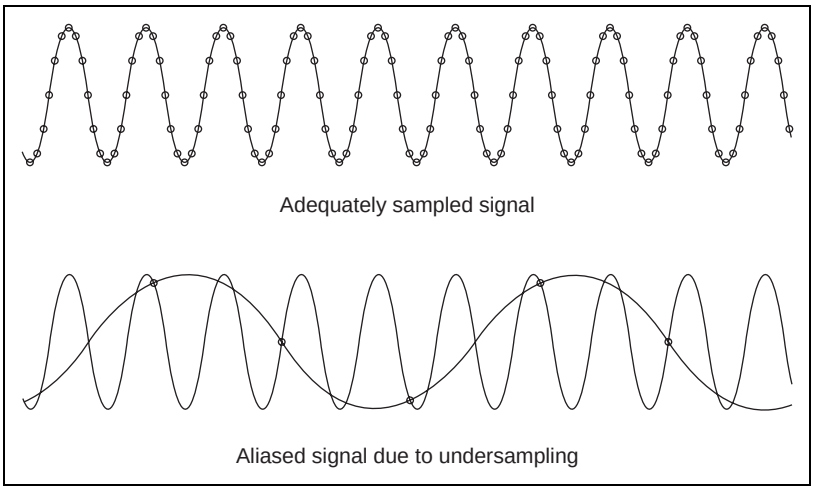


Figure 3. Adequate and Inadequate Signal Sampling

When the Nyquist criterion is violated, frequency components above half the sampling frequency appear as frequency components below half the sampling frequency, resulting in an erroneous representation of the signal. For example, a component at frequency

$$\frac{F_s}{2} < f_0 < F_s$$

appears as the frequency $F_s - f_0$.

Figure 4 shows the alias frequencies that appear when the signal with real components at 25, 70, 160, and 510 Hz is sampled at 100 Hz. Alias frequencies appear at 10, 30, and 40 Hz.

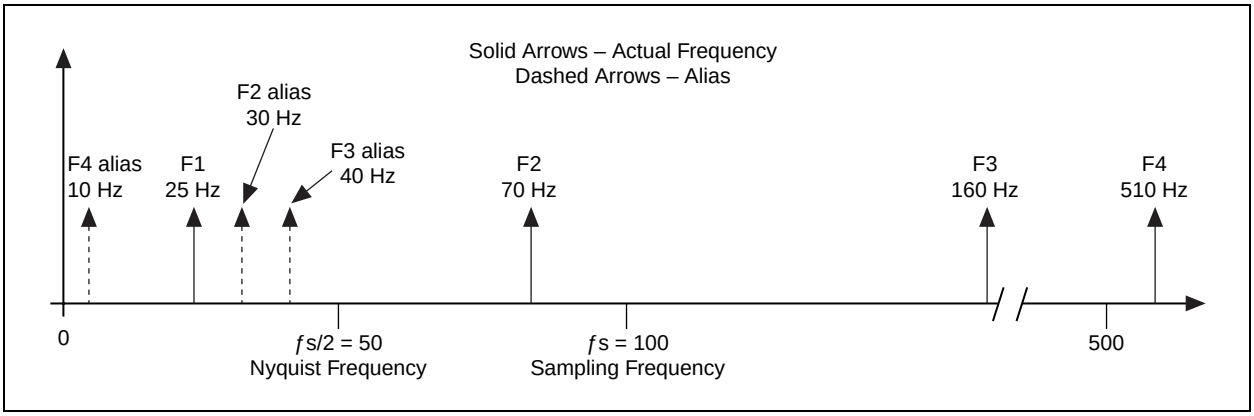


Figure 4. Alias Frequencies Resulting from Sampling a Signal at 100 Hz That Contains Frequency Components Greater than or Equal to 50 Hz

Before a signal is digitized, you can prevent aliasing by using antialiasing filters to attenuate the frequency components at and above half the sampling frequency to a level below the dynamic range of the analog-to-digital converter (ADC). For example, if the digitizer has a full-scale range of 80 dB, frequency components at and above half the sampling frequency must be attenuated to 80 dB below full scale.

These higher frequency components, do not interfere with the measurement. If you know that the frequency bandwidth of the signal being measured is lower than half the sampling frequency, you can choose not to use an antialiasing filter. Figure 5 shows the input frequency response of the National Instruments PCI-4450 Family dynamic signal acquisition boards, which have antialiasing filters. Note how an input signal at or above half the sampling frequency is severely attenuated.

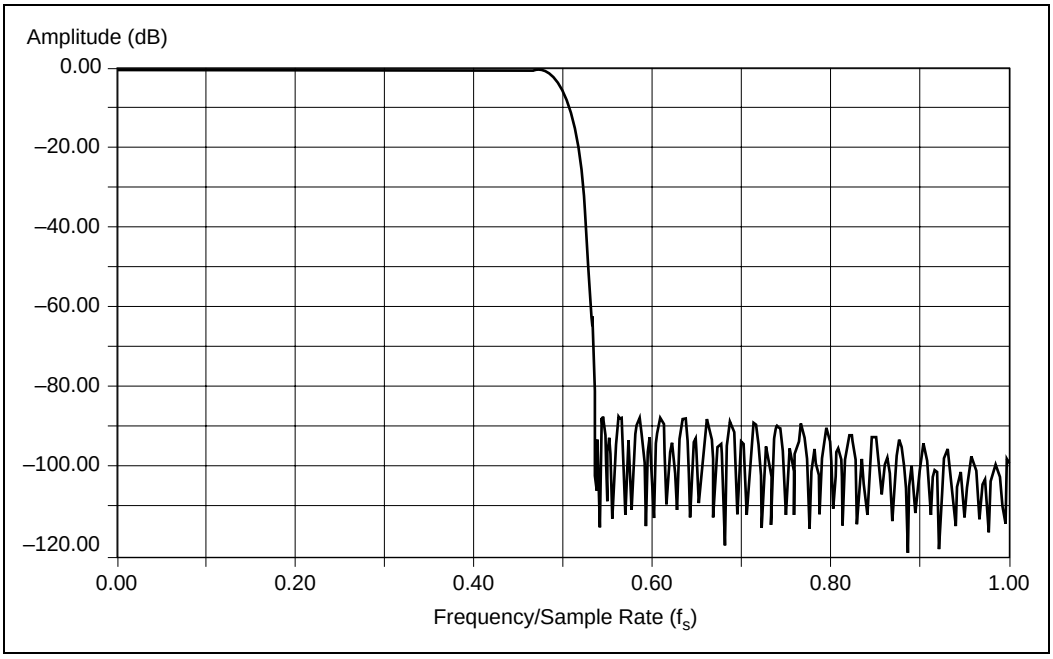


Figure 5. Bandwidth of PCI-4450 Family Input Versus Frequency, Normalized to Sampling Rate

Limitations of the Acquisition Front End

In addition to reducing frequency components greater than half the sampling frequency, the acquisition front end you use introduces some bandwidth limitations below half the sampling frequency. To eliminate signals at or above half of the sampling rate to less than the measurement range, antialiasing filters start to attenuate frequencies at some point below half the sampling rate. Because these filters attenuate the highest frequency portion of the spectrum, typically you want to limit the plot to the bandwidth you consider valid for the measurement.

For example, in the case of the PCI-4450 Family sample shown in Figure 5, amplitude flatness is maintained to within ±0.1 dB, at up to 0.464 of the sampling frequency to 20 kHz for all gain settings, +1 dB to 95 kHz, and then the input gain starts to attenuate. The –3 dB point (or half-power bandwidth) of the input occurs at 0.493 of the input spectrum. Therefore, instead of showing the input spectrum all the way out to half the sampling frequency, you may want to show only 0.464 of the input spectrum. To do this, multiply the number of points acquired by 0.464, respectively, to compute the number of frequency lines to display.

The characteristics of the signal acquisition front end affect the measurement. The National Instruments PCI-4450 Family dynamic signal acquisition boards and the NI 4551 and NI 4552 dynamic signal analyzers are excellent acquisition front ends for performing FFT-based signal analysis measurements. These boards use delta-sigma modulation technology, which yields excellent amplitude flatness, high-performance antialiasing filters, and wide dynamic range as shown in Figure 5. The input channels are also simultaneously sampled for good multichannel measurement performance.

At a sampling frequency of 51.2 kHz, these boards can perform frequency measurements in the range of DC to 23.75 kHz. Amplitude flatness is ±0.1 dB maximum from DC to 23.75 kHz. Refer to the *PCI-4451/4452/4453/4454 User Manual* for more information about these boards.

Calculating the Measurement Bandwidth or Number of Lines for a Given Sampling Frequency

The dynamic signal acquisition boards have antialiasing filters built into the digitizing process. In addition, the cutoff filter frequency scales with the sampling rate to meet the Nyquist criterion as shown in Figure 5. The fast cutoff of the antialiasing filters on these boards means that the number of useful frequency lines in a 1,024-point FFT-based spectrum is 475 lines for ±0.1 dB amplitude flatness.

To calculate the measurement bandwidth for a given sampling frequency, multiply the sampling frequency by 0.464 for the ±0.1 dB flatness. Also, the larger the FFT, the larger the number of frequency lines. A 2,048-point FFT yields twice the number of lines listed above. Contrast this with typical benchtop instruments, which have 400 or 800 useful lines for a 1,024- point or 2,048-point FFT, respectively.

Dynamic Range Specifications

The signal-to-noise ratio (SNR) of the PCI-4450 Family boards is 93 dB. SNR is defined as

$$\text{SNR} = 10\log_{10}\left(\frac{V_s^2}{V_n^2}\right)\text{dB}$$

where V_s and V_n are the rms amplitudes of the signal and noise, respectively. A bandwidth is usually given for SNR. In this case, the bandwidth is the frequency range of the board input, which is related to the sampling rate as shown in Figure 5. The 93 dB SNR means that you can detect the frequency components of a signal that is as small as 93 dB below the full-scale range of the board. This is possible because the total input noise level caused by the acquisition front end is 93 dB below the full-scale input range of the board.

If the signal you monitor is a narrowband signal (that is, the signal energy is concentrated in a narrow band of frequencies), you are able to detect an even lower level signal than –93 dB. This is possible because the noise energy of the board is spread out over the entire input frequency range. Refer to the *Computing Noise Level and Power Spectral Density* section later in this application note for more information about narrowband versus broadband levels.

The spurious-free dynamic range of the dynamic signal acquisition boards is 95 dB. Besides input noise, the acquisition front end may introduce spurious frequencies into a measured spectrum because of harmonic or intermodulation distortion, among other things. This 95 dB level indicates that any such spurious frequencies are at least 95 dB below the full-scale input range of the board.

The signal-to-total-harmonic-distortion (THD)-plus-noise ratio, which excludes intermodulation distortion, is 90 dB from 0 to 20 kHz. THD is a measure of the amount of distortion introduced into a signal because of the nonlinear behavior of the acquisition front end. This harmonic distortion shows up as harmonic energy added to the spectrum for each of the discrete frequency components present in the input signal.

The wide dynamic range specifications of these boards is largely due to the 16-bit resolution ADCs. Figure 6 shows a typical spectrum plot of the PCI-4450 Family dynamic range with a full-scale 997 Hz signal applied. You can see that the harmonics of the 997 Hz input signal, the noise floor, and any other spurious frequencies are below 95 dB. In contrast, dynamic range specifications for benchtop instruments typically range from 70 dB to 80 dB using 12-bit and 13-bit ADC technology.

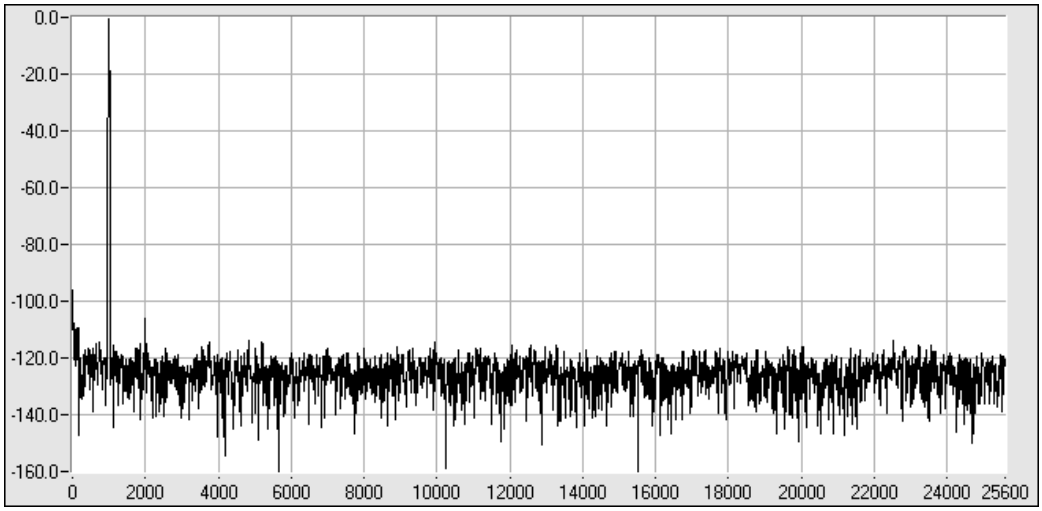


Figure 6. PCI-4450 Family Spectrum Plot with 997 Hz Input at Full Scale (Full Scale = 0 dB)

Using Windows Correctly

As mentioned in the Introduction, using windows correctly is critical to FFT-based measurement. This section describes the problem of spectral leakage, the characteristics of windows, some strategies for choosing windows, and the importance of scaling windows.

Spectral Leakage

For an accurate spectral measurement, it is not sufficient to use proper signal acquisition techniques to have a nicely scaled, single-sided spectrum. You might encounter spectral leakage. Spectral leakage is the result of an assumption in the FFT algorithm that the time record is exactly repeated throughout all time and that signals contained in a time record are thus periodic at intervals that correspond to the length of the time record. If the time record has a nonintegral number of cycles, this assumption is violated and spectral leakage occurs. Another way of looking at this case is that the nonintegral cycle frequency component of the signal does not correspond exactly to one of the spectrum frequency lines.

There are only two cases in which you can guarantee that an integral number of cycles are always acquired. One case is if you are sampling synchronously with respect to the signal you measure and can therefore deliberately take an integral number of cycles.

Another case is if you capture a transient signal that fits entirely into the time record. In most cases, however, you measure an unknown signal that is stationary; that is, the signal is present before, during, and after the acquisition. In this case, you cannot guarantee that you are sampling an integral number of cycles. Spectral leakage distorts the measurement in such a way that energy from a given frequency component is spread over adjacent frequency lines or bins. You can use windows to minimize the effects of performing an FFT over a nonintegral number of cycles.

Figure 7 shows the effects of three different windows — none (Uniform), Hanning (also commonly known as Hann), and Flat Top — when an integral number of cycles have been acquired, in this figure, 256 cycles in a 1,024-point record. Notice that the windows have a main lobe around the frequency of interest. This main lobe is a frequency domain characteristic of windows. The Uniform window has the narrowest lobe, and the Hann and Flat Top windows introduce some spreading. The Flat Top window has a broader main lobe than the others. For an integral number of cycles, all windows yield the same peak amplitude reading and have excellent amplitude accuracy.

Figure 7 also shows the values at frequency lines of 254 Hz through 258 Hz for each window. The amplitude error at 256 Hz is 0 dB for each window. The graph shows the spectrum values between 240 and 272 Hz. The actual values in the resulting spectrum array for each window at 254 through 258 Hz are shown below the graph. Δf is 1 Hz.

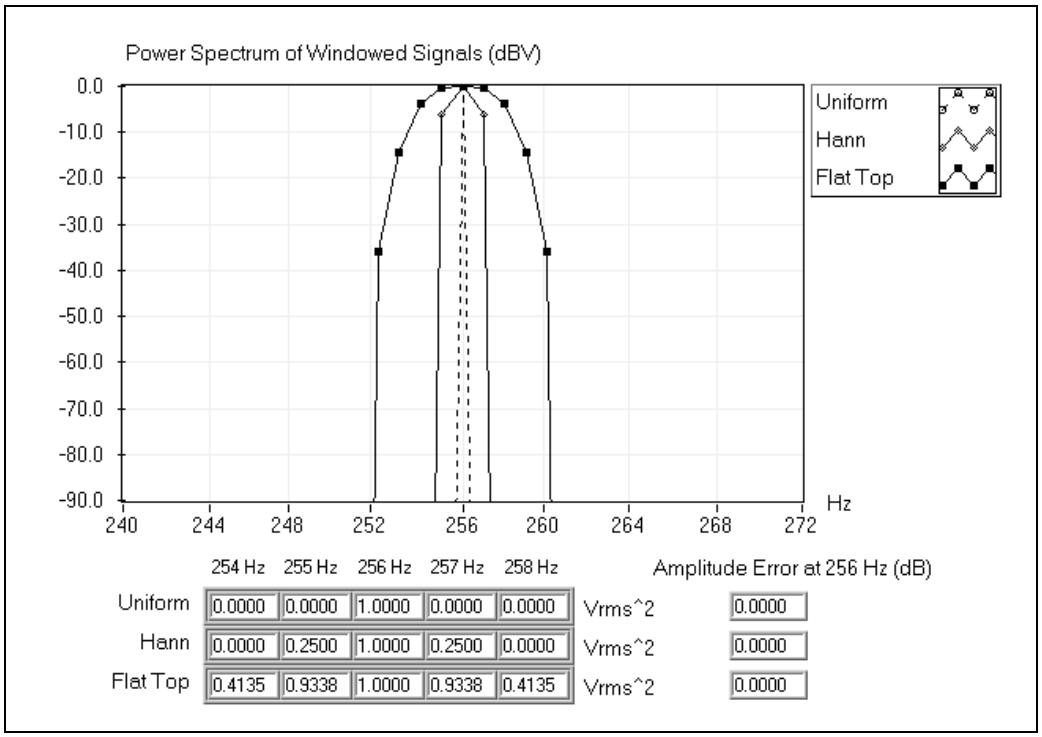


Figure 7. Power Spectrum of 1 Vrms Signal at 256 Hz with Uniform, Hann, and Flat Top Windows

Figure 8 shows the leakage effects when you acquire 256.5 cycles. Notice that at a nonintegral number of cycles, the Hann and Flat Top windows introduce much less spectral leakage than the Uniform window. Also, the amplitude error is better with the Hann and Flat Top windows. The Flat Top window demonstrates very good amplitude accuracy but also has a wider spread and higher side lobes than the Hann window.

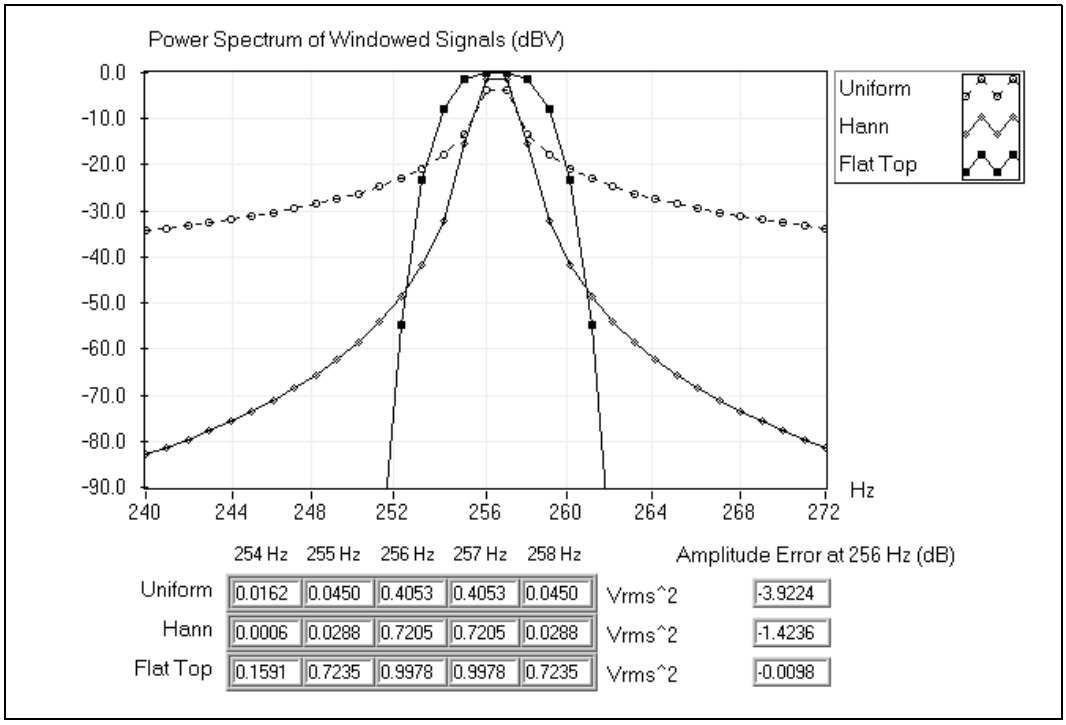


Figure 8. Power Spectrum of 1 Vrms Signal at 256.5 Hz with Uniform, Hann, and Flat Top Windows

In addition to causing amplitude accuracy errors, spectral leakage can obscure adjacent frequency peaks. Figure 9 shows the spectrum for two close frequency components when no window is used and when a Hann window is used.

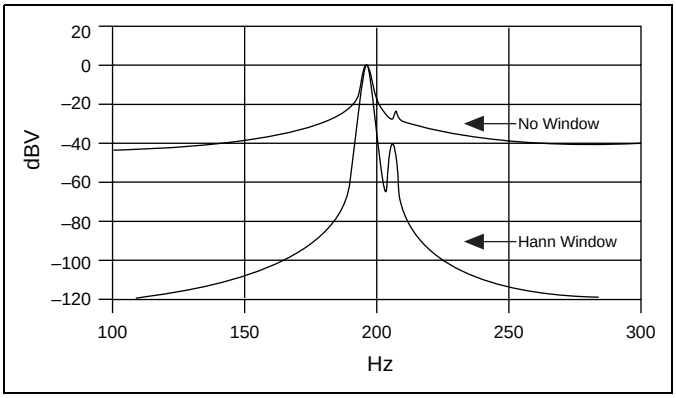


Figure 9. Spectral Leakage Obscuring Adjacent Frequency Components

Window Characteristics

To understand how a given window affects the frequency spectrum, you need to understand more about the frequency characteristics of windows. The windowing of the input data is equivalent to convolving the spectrum of the original signal with the spectrum of the window as shown in Figure 10. Even if you use no window, the signal is convolved with a rectangular-shaped window of uniform height, by the nature of taking a snapshot in time of the input signal. This convolution has a sine function characteristic spectrum. For this reason, no window is often called the Uniform or Rectangular window because there is still a windowing effect.

An actual plot of a window shows that the frequency characteristic of a window is a continuous spectrum with a main lobe and several side lobes. The main lobe is centered at each frequency component of the time-domain signal, and the side lobes approach zero at

$$\Delta f = \frac{F_s}{N} \text{ i}$$

intervals on each side of the main lobe.

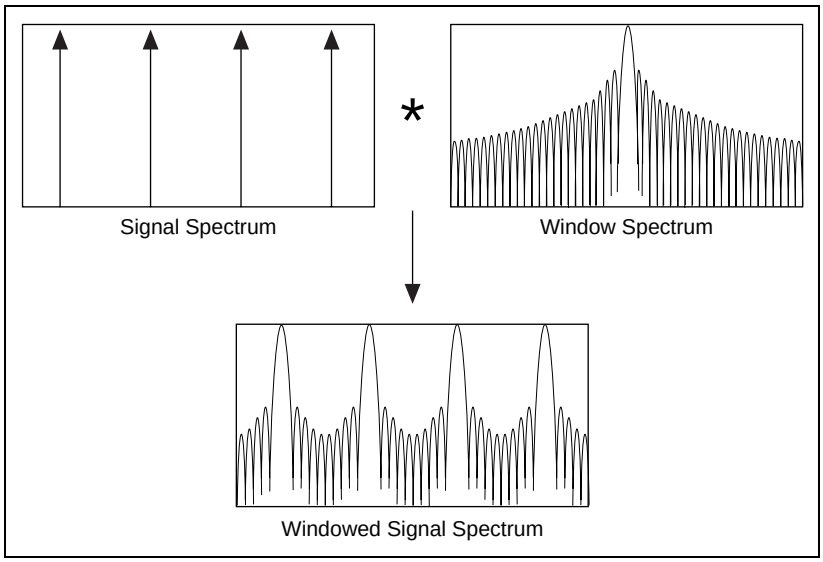


Figure 10. Frequency Characteristics of a Windowed Spectrum

An FFT produces a discrete frequency spectrum. The continuous, periodic frequency spectrum is sampled by the FFT, just as the time-domain signal was sampled by the ADC. What appears in each frequency line of the FFT is the value of the continuous convolved spectrum at each FFT frequency line. This is sometimes referred to as the picket-fence effect because the FFT result is analogous to viewing the continuous windowed spectrum through a picket fence with slits at intervals that corresponds to the frequency lines.

If the frequency components of the original signal match a frequency line exactly, as is the case when you acquire an integral number of cycles, you see only the main lobe of the spectrum. Side lobes do not appear because the spectrum of the window approaches zero at Δf intervals on either side of the main lobe. Figure 7 illustrates this case.

If a time record does not contain an integral number of cycles, the continuous spectrum of the window is shifted from the main lobe center at a fraction of Δf that corresponds to the difference between the frequency component and the FFT line frequencies. This shift causes the side lobes to appear in the spectrum. In addition, there is some amplitude error at the frequency peak, as shown in Figure 8, because the main lobe is sampled off center (the spectrum is smeared).

Figure 11 shows the frequency spectrum characteristics of a window in more detail. The side lobe characteristics of the window directly affect the extent to which adjacent frequency components bias (leak into) adjacent frequency bins. The side lobe response of a strong sinusoidal signal can overpower the main lobe response of a nearby weak sinusoidal signal.

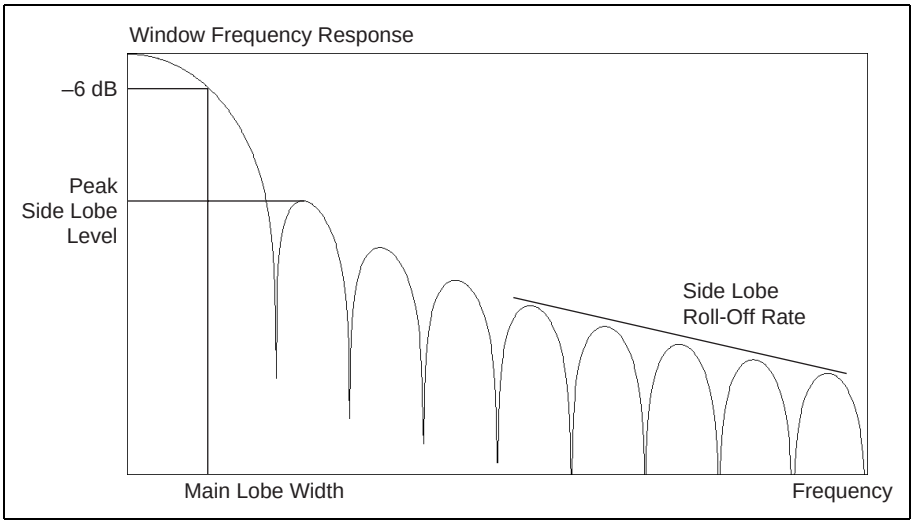


Figure 11. Frequency Response of a Window

Another important characteristic of window spectra is main lobe width. The frequency resolution of the windowed signal is limited by the width of the main lobe of the window spectrum. Therefore, the ability to distinguish two closely spaced frequency components increases as the main lobe of the window narrows. As the main lobe narrows and spectral resolution improves, the window energy spreads into its side lobes, and spectral leakage worsens. In general, then, there is a trade off between leakage suppression and spectral resolution.

Defining Window Characteristics

To simplify choosing a window, you need to define various characteristics so that you can make comparisons between windows. Figure 11 shows the spectrum of a typical window. To characterize the main lobe shape, the –3 dB and –6 dB main lobe width are defined to be the width of the main lobe (in FFT bins or frequency lines) where the window response becomes 0.707 (–3 dB) and 0.5 (–6 dB), respectively, of the main lobe peak gain.

To characterize the side lobes of the window, the maximum side lobe level and side lobe roll-off rate are defined. The maximum side lobe level is the level in decibels relative to the main lobe peak gain, of the maximum side lobe. The side lobe roll-off rate is the asymptotic decay rate, in decibels per decade of frequency, of the peaks of the side lobes. Table 1 lists the characteristics of several window functions and their effects on spectral leakage and resolution.

Table 1. Characteristics of Window Functions

Window	–3 dB Main Lobe Width (bins)	–6 dB Main Lobe Width (bins)	Maximum Side Lobe Level (dB)	Side Lobe Roll-Off Rate (dB/decade)
Uniform (None)	0.88	1.21	-13	20
Hanning (Hann)	1.44	2.00	-32	60
Hamming	1.30	1.81	-43	20
Blackman-Harris	1.62	2.27	-71	20
Exact Blackman	1.52	2.13	-67	20
Blackman	1.68	2.35	-58	60
Flat Top	2.94	3.56	-44	20

Strategies for Choosing Windows

Each window has its own characteristics, and different windows are used for different applications. To choose a spectral window, you must guess the signal frequency content. If the signal contains strong interfering frequency components distant from the frequency of interest, choose a window with a high side lobe roll-off rate. If there are strong interfering signals near the frequency of interest, choose a window with a low maximum side lobe level.

If the frequency of interest contains two or more signals very near to each other, spectral resolution is important. In this case, it is best to choose a window with a very narrow main lobe. If the amplitude accuracy of a single frequency component is more important than the exact location of the component in a given frequency bin, choose a window with a wide main lobe. If the signal spectrum is rather flat or broadband in frequency content, use the Uniform window (no window). In general, the Hann window is satisfactory in 95% of cases. It has good frequency resolution and reduced spectral leakage.

The Flat Top window has good amplitude accuracy, but because it has a wide main lobe, it has poor frequency resolution and more spectral leakage. The Flat Top window has a lower maximum side lobe level than the Hann window, but the Hann window has a faster roll off-rate. If you do not know the nature of the signal but you want to apply a window, start with the Hann window. Figures 7 and 8 contrast the characteristics of the Uniform, Hann, and Flat Top Windows windows.

If you are analyzing transient signals such as impact and response signals, it is better not to use the spectral windows because these windows attenuate important information at the beginning of the sample block. Instead, use the Force and Exponential windows. A Force window is useful in analyzing shock stimuli because it removes stray signals at the end of the signal. The Exponential window is useful for analyzing transient response signals because it damps the end of the signal, ensuring that the signal fully decays by the end of the sample block.

Selecting a window function is not a simple task. In fact, there is no universal approach for doing so. However, Table 2 can help you in your initial choice. Always compare the performance of different window functions to find the best one for the application. Refer to the references at the end of this application note for more information about windows.

Table 2. Initial Window Choice Based on Signal Content

Signal Content	Window
Sine wave or combination of sine waves	Hann
Sine wave (amplitude accuracy is important)	Flat Top
Narrowband random signal (vibration data)	Hann
Broadband random (white noise)	Uniform
Closely spaced sine waves	Uniform, Hamming
Excitation signals (Hammer blow)	Force
Response signals	Exponential
Unknown content	Hann

Scaling Windows

Windows are useful in reducing spectral leakage when using the FFT for spectral analysis. However, because windows are multiplied with the acquired time-domain signal, they introduce distortion effects of their own. The windows change the overall amplitude of the signal. The windows used to produce the plots in Figures 7 and 8 were scaled by dividing the windowed array by the coherent gain of the window. As a result, each window yields the same spectrum amplitude result within its accuracy constraints.

You can think of an FFT as a set of parallel filters, each Δf in bandwidth. Because of the spreading effect of a window, each window increases the effective bandwidth of an FFT bin by an amount known as the equivalent noise-power bandwidth of the window. The power of a given frequency peak is computed by adding the adjacent frequency bins around a peak and is inflated by the bandwidth of the window. You must take this inflation into account when you perform computations based on the spectrum. Refer to the Computations on the Spectrum section for sample computations.

Table 3 lists the scaling factor (or coherent gain), the noise power bandwidth, and the worst-case peak amplitude accuracy caused by off-center components for several popular windows.

Table 3. Correction Factors and Worst-Case Amplitude Errors for Windows

Window	Scaling Factor (Coherent Gain)	Noise Power Bandwidth	Worst-Case Amplitude Error (dB)
Uniform (none)	1.00	1.00	3.92
Hann	0.50	1.50	1.42
Hamming	0.54	1.36	1.75
Blackman-Harris	0.42	1.71	1.13
Exact Blackman	0.43	1.69	1.15
Blackman	0.42	1.73	1.10
Flat Top	0.22	3.77	< 0.01

Computations on the Spectrum

When you have the amplitude or power spectrum, you can compute several useful characteristics of the input signal, such as power and frequency, noise level, and power spectral density.

Estimating Power and Frequency

The preceding windowing examples demonstrate that if you have a frequency component in between two frequency lines, it appears as energy spread among adjacent frequency lines with reduced amplitude. The actual peak is between the two frequency lines. In Figure 8, the amplitude error at 256.5 Hz is due to the fact that the window is sampled at ±0.5 Hz around the center of its main lobe rather than at the center where the amplitude error would be 0. This is the picket-fence effect explained in the Window Characteristics section of this application note.

You can estimate the actual frequency of a discrete frequency component to a greater resolution than the Δf given by the FFT by performing a weighted average of the frequencies around a detected peak in the power spectrum.

$$\text{Estimated Frequency} = \frac{\sum_{i=j-3}^{j+3} (\text{Power}(i) \bullet i \bullet \Delta f)}{\sum_{i=j-3}^{j+3} \text{Power}(i)}$$

where j is the array index of the apparent peak of the frequency of interest and

$$\Delta f = \frac{F_s}{N}$$

The span j ±3 is reasonable because it represents a spread wider than the main lobes of the windows listed in Table 3.

Similarly, you can estimate the power in $Vrms^2$ of a given peak discrete frequency component by summing the power in the bins around the peak (computing the area under the peak)

$$\text{Estimated Power} = \frac{\sum_{i=j-3}^{j+3} \text{Power}(i)}{\text{noise power bandwidth of window}}$$

Notice that this method is valid only for a spectrum made up of discrete frequency components. It is not valid for a continuous spectrum. Also, if two or more frequency peaks are within six lines of each other, they contribute to inflating the estimated powers and skewing the actual frequencies. You can reduce this effect by decreasing the number of lines spanned by the preceding computations. If two peaks are that close, they are probably already interfering with one another because of spectral leakage.

Similarly, if you want the total power in a given frequency range, sum the power in each bin included in the frequency range and divide by the noise power bandwidth of the windows.

Computing Noise Level and Power Spectral Density

The measurement of noise levels depends on the bandwidth of the measurement. When looking at the noise floor of a power spectrum, you are looking at the narrowband noise level in each FFT bin. Thus, the noise floor of a given power spectrum depends on the Δf of the spectrum, which is in turn controlled by the sampling rate and number of points. In other words, the noise level at each frequency line reads as if it were measured through a Δf Hz filter centered at that frequency line. Therefore, for a given sampling rate, doubling the number of points acquired reduces the noise power that appears in each bin by 3 dB. Discrete frequency components theoretically have zero bandwidth and therefore do not scale with the number of points or frequency range of the FFT.

To compute the SNR, compare the peak power in the frequencies of interest to the broadband noise level. Compute the broadband noise level in $Vrms^2$ by summing all the power spectrum bins, excluding any peaks and the DC component, and dividing the sum by the equivalent noise bandwidth of the window. For example, in Figure 6 the noise floor appears to be more than 120 dB below full scale, even though the PCI-4450 Family dynamic range is only 93 dB. If you were to sum all the bins, excluding DC, and any harmonic or other peak components and divide by the noise power bandwidth of the window you used, the noise power level compared to full scale would be around –93 dB from full scale.

Because of noise-level scaling with Δf, spectra for noise measurement are often displayed in a normalized format called power or amplitude spectral density. This normalizes the power or amplitude spectrum to the spectrum that would be measured by a 1 Hz-wide square filter, a convention for noise-level measurements. The level at each frequency line then reads as if it were measured through a 1 Hz filter centered at that frequency line.

Power spectral density is computed as

$$\text{Power spectral density} = \frac{\text{Power Spectrum in } V_{rms}^2}{\Delta f \times \text{Noise Power Bandwidth of Window}}$$

The units are then in $\frac{V_{rms}^2}{Hz}$ or $\frac{V^2}{Hz}$

Amplitude spectral density is computed as:

$$\text{Amplitude Spectral Density} = \frac{\text{Amplitude Spectrum in } V_{rms}}{\sqrt{\Delta f \times \text{Noise Power Bandwidth of Window}}}$$

The units are then in $\frac{V_{rms}}{\sqrt{Hz}}$ or $\frac{V}{\sqrt{Hz}}$.

The spectral density format is appropriate for random or noise signals but inappropriate for discrete frequency components because the latter theoretically have zero bandwidth.

FFT-Based Network Measurement

When you understand how to handle computations with the FFT and power spectra, and you understand the influence of windows on the spectrum, you can compute several FFT-based functions that are extremely useful for network analysis. These include the transfer, impulse, and coherence functions. Refer to the Frequency Response and Network Analysis section of this application note for more information about these functions. Refer to the Signal Sources for Frequency Response Measurement section for more information about Chirp signals and broadband noise signals.

Cross Power Spectrum

One additional building block is the cross power spectrum. The cross power spectrum is not typically used as a direct measurement but is an important building block for other measurements.

The two-sided cross power spectrum of two time-domain signals A and B is computed as

$$\text{Cross Power Spectrum } S_{AB}(f) = \frac{FFT(B) \times FFT^*(A)}{N^2}$$

The cross power spectrum is in two-sided complex form. To convert to magnitude and phase, use the Rectangular-To-Polar conversion function. To convert to a single-sided form, use the same method described in the Converting from a Two-Sided Power Spectrum to a Single-Sided Power Spectrum section of this application note. The units of the single-sided form are in volts (or other quantity) rms squared.

The power spectrum is equivalent to the cross power spectrum when signals A and B are the same signal. Therefore, the power spectrum is often referred to as the auto power spectrum or the auto spectrum. The single-sided cross power spectrum yields the product of the rms amplitudes of the two signals, A and B, and the phase difference between the two signals.

When you know how to use these basic blocks, you can compute other useful functions, such as the Frequency Response function.

Frequency Response and Network Analysis

Three useful functions for characterizing the frequency response of a network are the transfer, impulse response, and coherence functions.

The frequency response of a network is measured by applying a stimulus to the network as shown in Figure 12 and computing the transfer function from the stimulus and response signals.

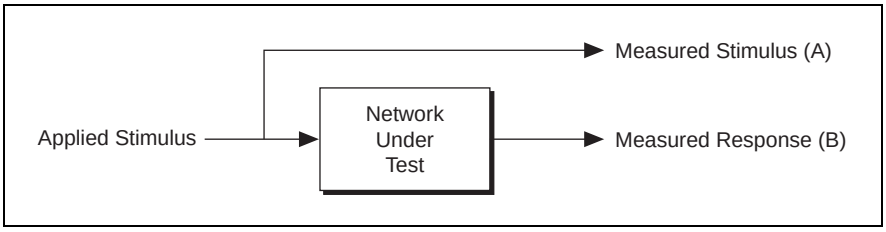


Figure 12. Configuration for Network Analysis

Transfer Function

The transfer function gives the gain and phase versus frequency of a network and is typically computed as

$$\text{Transfer Function } H(f) = \frac{\text{Cross Power Spectrum (Stimulus, Response)}}{\text{Power Spectrum (Stimulus)}} = \frac{S_{AB}(f)}{S_{AA}(f)}$$

where A is the stimulus signal and B is the response signal.

The transfer function is in two-sided complex form. To convert to the frequency response gain (magnitude) and the frequency response phase, use the Rectangular-To-Polar conversion function. To convert to single-sided form, simply discard the second half of the array.

You may want to take several transfer function readings and then average them. To do so, average the cross power spectrum, $S_{AB}(f)$, by summing it in the complex form then dividing by the number of averages, before converting it to magnitude and phase, and so forth. The power spectrum, $S_{AA}(f)$, is already in real form and is averaged normally.

Impulse Response Function

The impulse response function of a network is the time-domain representation of the transfer function of the network. It is the output time-domain signal generated when an impulse is applied to the input at time t = 0.

To compute the impulse response of the network, take the inverse FFT of the two-sided complex transfer function as described in the *Transfer Function* section of this application note.

$$\text{Impulse Response (f)} = \text{Inverse FFT (Transfer Function H(f))} = \text{Inverse FFT}\left(\frac{S_{AB}(f)}{S_{AA}(f)}\right)$$

The result is a time-domain function. To average multiple readings, take the inverse FFT of the averaged transfer function.

Coherence Function

The coherence function is often used in conjunction with the transfer function as an indication of the quality of the transfer function measurement and indicates how much of the response energy is correlated to the stimulus energy. If there is another signal present in the response, either from excessive noise or from another signal, the quality of the network response measurement is poor. You can use the coherence function to identify both excessive noise and causality, that is, identify which of the multiple signal sources are contributing to the response signal. The coherence function is computed as

$$\text{Coherence Function (f)} = \frac{[\text{Magnitude}(\text{Averaged } S_{AB}(f))]^2}{\text{Averaged } S_{AA}(f) \bullet \text{Averaged } S_{BB}(f)}$$

The result is a value between zero and one versus frequency. A zero for a given frequency line indicates no correlation between the response and the stimulus signal. A one for a given frequency line indicates that the response energy is 100 percent due to the stimulus signal; in other words, there is no interference at that frequency.

For a valid result, the coherence function requires an average of two or more readings of the stimulus and response signals. For only one reading, it registers unity at all frequencies. To average the cross power spectrum, $S_{AB}(f)$, average it in the complex form then convert to magnitude and phase as described in the Transfer Function section of this application note. The auto power spectra, $S_{AA}(f)$ and $S_{BB}(f)$, are already in real form, and you average them normally.

Signal Sources for Frequency Response Measurements

To achieve a good frequency response measurement, significant stimulus energy must be present in the frequency range of interest. Two common signals used are the chirp signal and a broadband noise signal. The chirp signal is a sinusoid swept from a start frequency to a stop frequency, thus generating energy across a given frequency range. White and pseudorandom noise have flat broadband frequency spectra; that is, energy is present at all frequencies.

It is best not to use windows when analyzing frequency response signals. If you generate a chirp stimulus signal at the same rate you acquire the response, you can match the acquisition frame size to match the length of the chirp. No window is generally the best choice for a broadband signal source. Because some stimulus signals are not constant in frequency across the time record, applying a window may obscure important portions of the transient response.

Conclusion

There are many issues to consider when analyzing and measuring signals from plug-in DAQ devices. Unfortunately, it is easy to make incorrect spectral measurements. Understanding the basic computations involved in FFT-based measurement, knowing how to prevent antialiasing, properly scaling and converting to different units, choosing and using windows correctly, and learning how to use FFT-based functions for network measurement are all critical to the success of analysis and measurement tasks. Being equipped with this knowledge and using the tools discussed in this application note can bring you more success with your individual application.

References

Harris, Fredric J. "On the Use of Windows for Harmonic Analysis with the Discrete Fourier Transform" in *Proceedings of the IEEE* Vol. 66, No. 1, January 1978.

Audio Frequency Fourier Analyzer (AFFA) User Guide, National Instruments, September 1991.

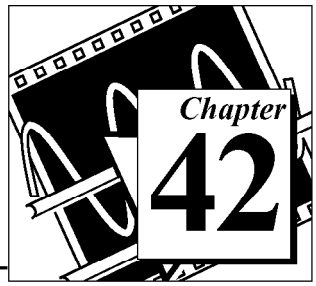
Horowitz, Paul, and Hill, Winfield, *The Art of Electronics*, 2nd Edition, Cambridge University Press, 1989.

Nuttall, Albert H. "Some Windows with Very Good Sidelobe Behavior," *IEEE Transactions on Acoustics, Speech, and Signal Processing* Vol. 29, No. 1, February 1981.

Randall, R.B., and Tech, B. *Frequency Analysis*, 3rd Edition, Bruël and Kjær, September 1979.

The Fundamentals of Signal Analysis, Application Note 243, Hewlett-Packard, 1985.

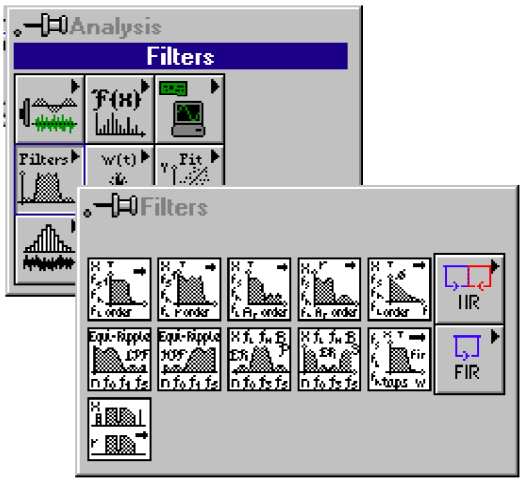
340555B-01 Jul00



Analysis Filter VIs

This chapter contains a brief discussion of digital filter theory and describes the VIs that implement IIR, FIR, and nonlinear filters.

To access the Analysis Filter palette, select **Function»Analysis»Filters**. The following illustration shows the options that are available on the Filter palette.



For examples of how to use the filter VIs, see the examples located in `examples\analysis\fltrxmpl.llb`.

Introduction to Digital Filtering Functions

Analog filter design is one of the most important areas of electronic design. Although analog filter design books featuring simple, tested filter designs exist, filter design is often reserved for specialists because it requires advanced mathematical knowledge and understanding of the processes involved in the system affecting the filter.

Modern sampling and digital signal processing tools have made it possible to replace analog filters with digital filters in applications that require flexibility and programmability. These applications include audio, telecommunications, geophysics, and medical monitoring.

Digital filters have the following advantages over their analog counterparts:

- They are software programmable.
- They are stable and predictable.
- They do not drift with temperature or humidity or require precision components.
- They have a superior performance-to-cost ratio.

You can use digital filters in LabVIEW to control parameters such as filter order, cutoff frequencies, amount of ripple, and stopband attenuation.

The digital filter VIs described in this section follow the virtual instrument philosophy. The VIs handle all the design issues, computations, memory management, and actual data filtering internally, and are transparent to the user. You do not have to be an expert in digital filters or digital filter theory to process the data.

The following discussion of sampling theory is intended to give you a better understanding of the filter parameters and how they relate to the input parameters.

The sampling theorem states that you can reconstruct a continuous-time signal from discrete, equally spaced samples if the sampling frequency is at least twice that of the highest frequency in the time signal. Assume you can sample the time signal of interest at Δt equally spaced intervals without losing information. The Δt parameter is the sampling interval.

You can obtain the sampling rate or sampling frequency f_s from the sampling interval

$$f_s = \frac{1}{\Delta t},$$

which means that, according to the sampling theorem, the highest frequency that the digital system can process is

$$f_{Nyq} = \frac{f_s}{2}.$$

The highest frequency the system can process is known as the Nyquist frequency. This also applies to digital filters. For example, if your sampling interval is

Δt = 0.001 sec,

then the sampling frequency is

f_s = 1000 IIz,

and the highest frequency that the system can process is

f_{Nyq} = 500 Hz.

The following types of filtering operations are based upon filter design techniques:

- Smoothing windows
- Infinite impulse response (IIR) or recursive digital filters
- Finite impulse response (FIR) or nonrecursive digital filters
- Nonlinear filters

The rest of this chapter presents a brief theoretical background on the IIR, FIR, and nonlinear techniques and discusses the digital filter VIs corresponding to each technique. Refer to Chapter 43, *Window VIs*, for information about the VIs that implement smoothing windows.

Infinite Impulse Response Filters

Infinite impulse response filters (IIR) are digital filters with impulse responses that can theoretically be infinite in length (duration). The general difference equation characterizing IIR filters is

$$y_i = \frac{1}{a_0}\left(\sum_{j=0}^{N_b-1} b_j x_{i-j} - \sum_{k=1}^{N_a-1} a_k y_{i-k}\right) \qquad (42\text{-}1)$$

where N_b is the number of *forward* coefficients (b_j) and N_a is the number of *reverse* coefficients (a_k).

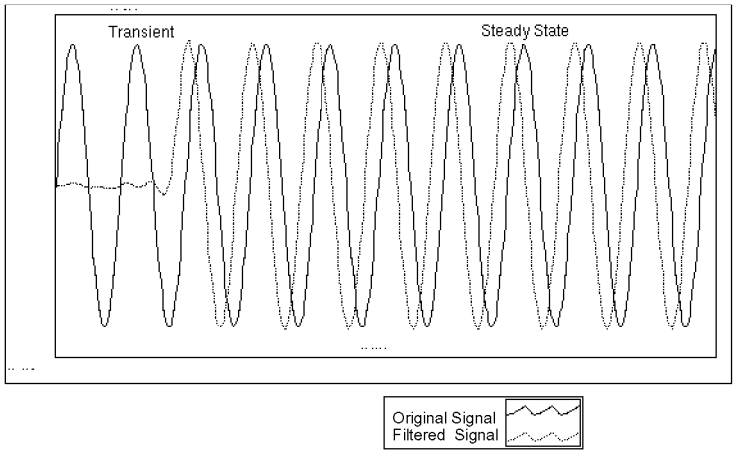
In most IIR filter designs (and in all of the LabVIEW IIR filters), coefficient a_0 is 1. The output sample at the present sample index i is the sum of scaled present and past inputs (x_i and x_{i-j} when $\neq 0$) and scaled past outputs (y_{i-k}). Because of this, IIR filters are also known as recursive filters or autoregressive moving-average (ARMA) filters.

The response of the general IIR filter to an impulse ($x_0 = 1$ and $x_i = 0$ for all $i \neq 0$) is called the impulse response of the filter. The impulse response of the filter described by equation (42-1) is indeed of infinite length for nonzero coefficients. In practical filter applications, however, the impulse response of stable IIR filters decays to near zero in a finite number of samples.

IIR filters in LabVIEW contain the following properties:

- Negative indices resulting from equation (42-1) are assumed to be zero the first time you call the VI.
- Because the initial filter state is assumed to be zero (negative indices), a transient proportional to the filter order occurs before the filter reaches a steady state. The duration of the transient response, or delay, for lowpass and highpass filters is equal to the filter order.
- Delay = order.
- The duration of the transient response for bandpass and bandstop filters is twice the filter order
- Delay = 2 * order.

You can eliminate this transient response on successive calls by enabling state memory. To enable state memory, set the **init/cont** control of the VI to TRUE (continuous filtering).



The number of elements in the filtered sequence equals the number of elements in the input sequence.

The filter retains the internal filter state values when the filtering completes.

The advantage of digital IIR filters over finite impulse response (FIR) filters is that IIR filters usually require fewer coefficients to perform similar filtering operations. Thus, IIR filters execute much faster and do not require extra memory, because they execute in place.

The disadvantage of IIR filters is that the phase response is nonlinear. If the application does not require phase information, such as simple signal monitoring, IIR filters may be appropriate. You should use FIR filters for those applications requiring linear phase responses.

Cascade Form IIR Filtering

Filters implemented using the structure defined by equation (42-2) directly are known as *direct form* IIR filters. Direct form implementations are often sensitive to errors introduced by coefficient quantization and by computational, precision limits. Additionally, a filter designed to be stable can become unstable with increasing coefficient length, which is proportional to filter order.

A less sensitive structure can be obtained by breaking up the direct form transfer function into lower order sections, or filter stages. The direct

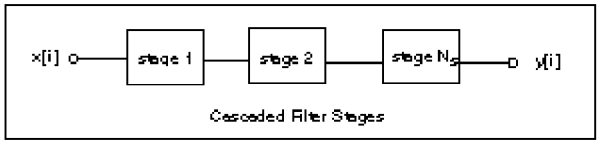
form transfer function of the filter given by equation (42-2) (with a_0 = 1) can be written as a ratio of z transforms, as follows:

$$H(z) = \frac{b_0 + b_1 z^{-1} + \ldots + b_{N_b - 1} z^{-(N_b - 1)}}{1 + a_1 z^{-1} + \ldots + a_{N_a - 1} z^{-(N_a - 1)}} \quad . \qquad (42\text{-}2)$$

By factoring equation (42-2) into second-order sections, the transfer function of the filter becomes a product of second-order filter functions

$$H(z) = \prod_{k=1}^{N_s} \frac{b_{0k} + b_{1k} z^{-1} + b_{2k} z^{-2}}{1 + a_{1k} z^{-1} + a_{2k} z^{-2}} \qquad (42\text{-}3)$$

where $N_s = \lfloor N_a/2 \rfloor$ is the largest integer $\leq N_a/2$, and $N_a \geq Nb$. (N_s is the *number of stages.*) This new filter structure can be described as a *cascade* of second-order filters.



Each individual stage is implemented using the *direct form II* filter structure because it requires a minimum number of arithmetic operations and a minimum number of delay elements (internal filter states). Each stage has one input, one output, and two past internal states ($s_k[i\text{-}1]$ and $s_k[i\text{-}2]$).

If n is the number of samples in the input sequence, the filtering operation proceeds as in the following equations:

$y_0[i] = x[i]$,

$s_k[i] = y_{k-1}[i-1] - a_{1k} s_k[i-1] - a_{2k} s_k[i\text{-}2], \quad k = 1, 2, \ldots, N_s$

$y_k[i] = b_{Ok} s_k[i] + b_{1k} s_k[i\text{-}1] + b_{2k} s_k[i\text{-}2], \; k = 1, 2, \ldots, N_s$

$y[i] = y_{Ns}[i]$

for each sample $i = 0, 1, 2, \ldots, n\text{-}1$.

For filters with a single cutoff frequency (lowpass and highpass), second-order filter stages can be designed directly. The overall IIR lowpass or highpass filter contains cascaded second-order filters.

For filters with two cutoff frequencies (bandpass and bandstop), fourth-order filter stages are a more natural form. The overall IIR bandpass or bandstop filter is cascaded fourth-order filters. The filtering operation for fourth-order stages proceeds as in the following equations:

$$y_0[i] = x[i],$$

$$s_k[i] = y_{k-1}[i-1] - a_{1k}s_k[i-1] - a_{2k}s_k[i-2] - a_{3k}s_k[i-3] - a_{4k}s_k[i-4],$$
$$k = 1, 2, \ldots, N_s$$

$$y_k[i] = b_{0k}s_k[i] + b_{1k}s_k[i-1] + b_{2k}s_k[i-2] + b_{3k}s_k[i-3] + b_{4k}s_k[i-4],$$
$$k = 1, 2, \ldots, N_s$$

$$y[i] = y_{Ns}[i].$$

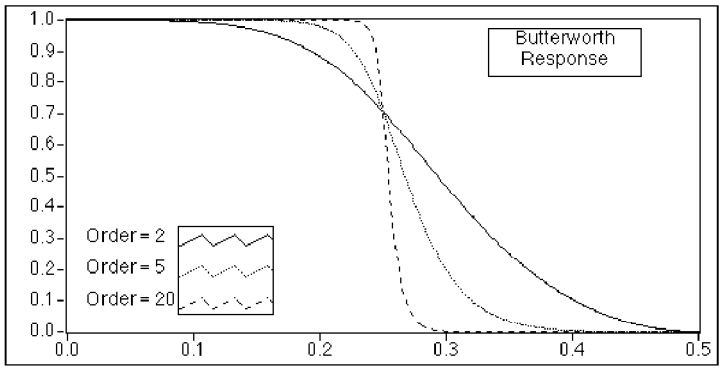
Notice that in the case of fourth-order filter stages, $N_s = \lfloor (N_a + 1)/4 \rfloor$.

Butterworth Filters

A smooth response at all frequencies a nd a monotonic decrease from the specified cutoff frequencies characterize the frequency response of Butterworth filters. Butterworth filters are maximally flat—the ideal response of unity in the passband and zero in the stopband. The half power frequency or the 3-dB down frequency corresponds to the specified cutoff frequencies.

The following illustration shows the response of a lowpass Butterworth filter. The advantage of Butterworth filters is a smooth, monotonically decreasing frequency response. After you set the cutoff frequency, LabVIEW sets the *steepness* of the transition proportional to the filter

order. Higher order Butterworth filters approach the ideal lowpass filter response.

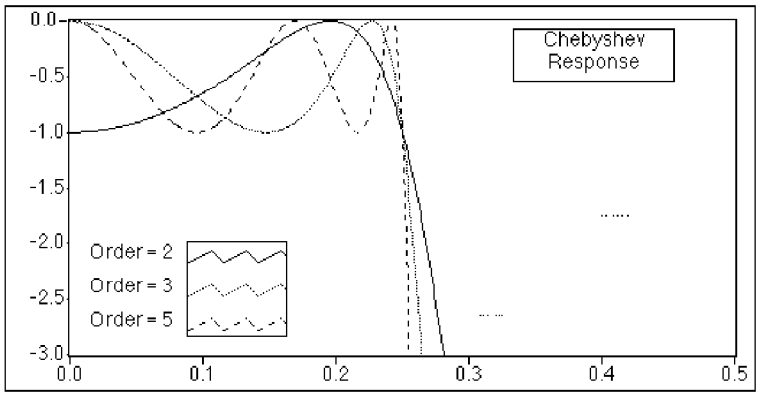


Chebyshev Filters

Butterworth filters do not always provide a good approximation of the ideal filter response because of the slow rolloff between the passband (the portion of interest in the spectrum) and the stopband (the unwanted portion of the spectrum).

Chebyshev filters minimize peak error in the passband by accounting for the maximum absolute value of the difference between the ideal filter and the filter response you want (the maximum tolerable error in the passband). The frequency response characteristics of Chebyshev filters have an equiripple magnitude response in the passband, monotonically decreasing magnitude response in the stopband, and a sharper rolloff than Butterworth filters.

The following graph shows the response of a lowpass Chebyshev filter. Notice that the equiripple response in the passband is constrained by the maximum tolerable ripple error and that the sharp rolloff appears in the stopband. The advantage of Chebyshev filters over Butterworth filters is that Chebyshev filters have a sharper transition between the passband and the stopband with a lower order filter. This produces smaller absolute errors and higher execution speeds.



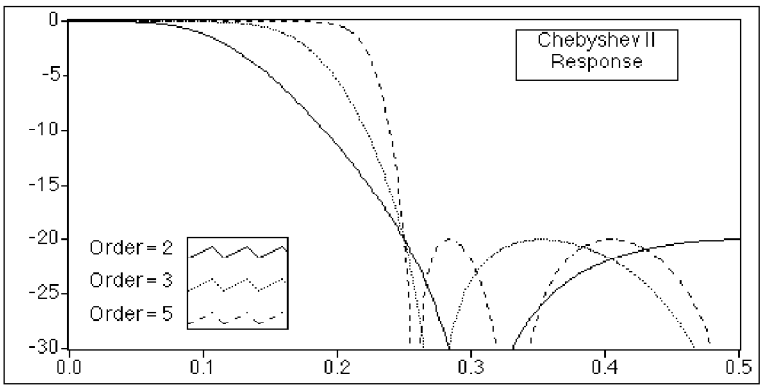
Chebyshev II or Inverse Chebyshev Filters

Chebyshev II, also known as inverse Chebyshev or Type II Chebyshev filters, are similar to Chebyshev filters, except that Chebyshev II filters distribute the error over the stopband (as opposed to the passband), and Chebyshev II filters are maximally flat in the passband (as opposed to the stopband).

Chebyshev II filters minimize peak error in the stopband by accounting for the maximum absolute value of the difference between the ideal filter and the filter response you want. The frequency response characteristics of Chebyshev II filters are equiripple magnitude response in the stopband, monotonically decreasing magnitude response in the passband, and a rolloff sharper than Butterworth filters.

The following graph plots the response of a lowpass Chebyshev II filter. Notice that the equiripple response in the stopband is constrained by the maximum tolerable error and that the smooth monotonic rolloff appears in the stopband. The advantage of Chebyshev II filters over Butterworth filters is that Chebyshev II filters give a sharper transition between the passband and the stopband with a lower order filter. This difference corresponds to a smaller, absolute error and higher execution speed. One advantage of Chebyshev II filters over regular Chebyshev filters is

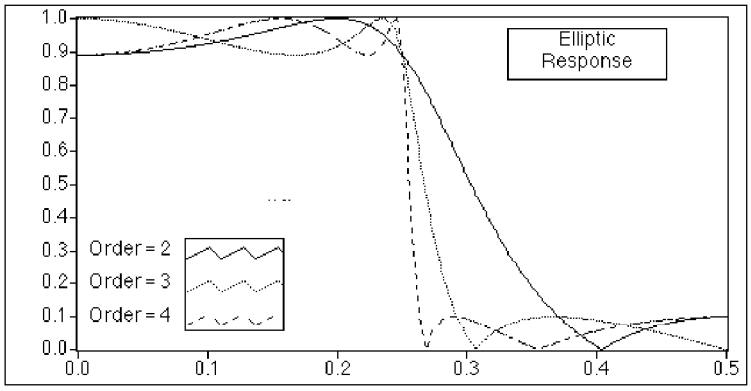
that Chebyshev II filters distribute the error in the stopband instead of the passband.



Elliptic (or Cauer) Filters

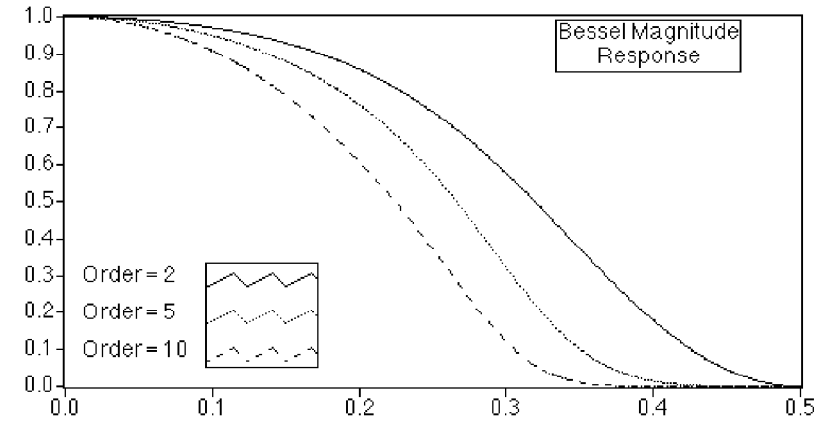
Elliptic filters minimize the peak error by distributing it over the passband and the stopband. Equi-ripples in the passband and the stopband characterize the magnitude response of elliptic filters. Compared with the same order Butterworth or Chebyshev filters, the elliptic design provides the sharpest transition between the passband and the stopband. For this reason, elliptic filters are used widely.

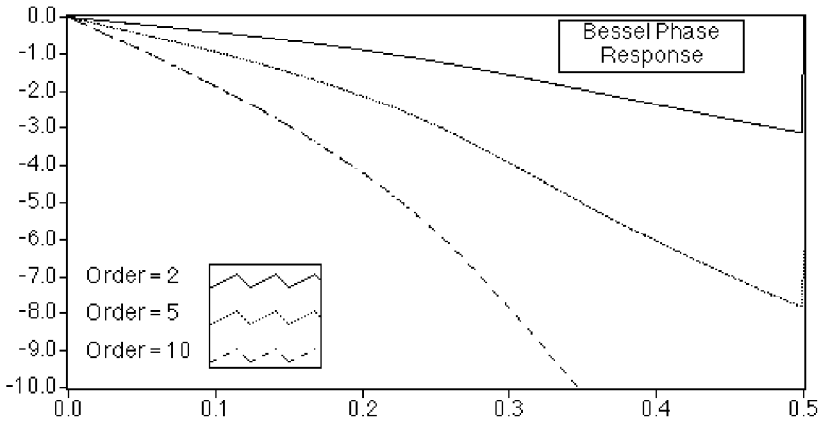
The following graph plots the response of a lowpass elliptic filter. Notice that the ripple in both the passband and stopband is constrained by the same maximum tolerable error (as specified by ripple amount in dB). Also, notice the sharp transition edge for even low-order elliptic filters.



Bessel Filters

You can use Bessel filters to reduce nonlinear phase distortion inherent in all IIR filters. In higher order filters and those with a steeper rolloff, this condition is more pronounced, especially in the transition regions of the filters. Bessel filters have maximally flat response in both magnitude and phase. Furthermore, the phase response in the passband of Bessel filters, which is the region of interest, is nearly linear. Like Butterworth filters, Bessel filters require high-order filters to minimize the error and, for this reason, are not widely used. You can also obtain linear phase response using FIR filter designs.The following graphs plot the response of a lowpass Bessel filter. Notice that the response is smooth at all frequencies, as well as monotonically decreasing in both magnitude and phase. Also, notice that the phase in the passband is nearly linear.





Finite Impulse Response Filters

Finite impulse response (FIR) filters are digital filters, which have a finite impulse response. FIR filters are also known as nonrecursive filters, convolution filters, or moving-average (MA) filters because you can express the output of an FIR filter as a finite convolution

$$y_i = \sum_{k=0}^{n-1} h_k x_{i-k}$$

where x represents the input sequence to be filtered, y represents the output filtered sequence, and h represents the FIR filter coefficients.

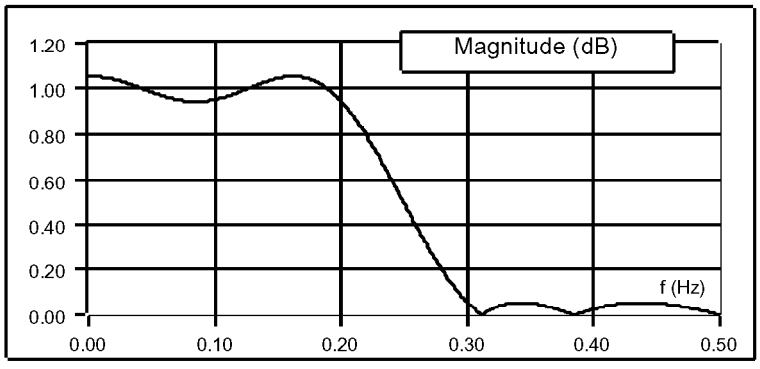
The following list gives the most important characteristics of FIR filters:

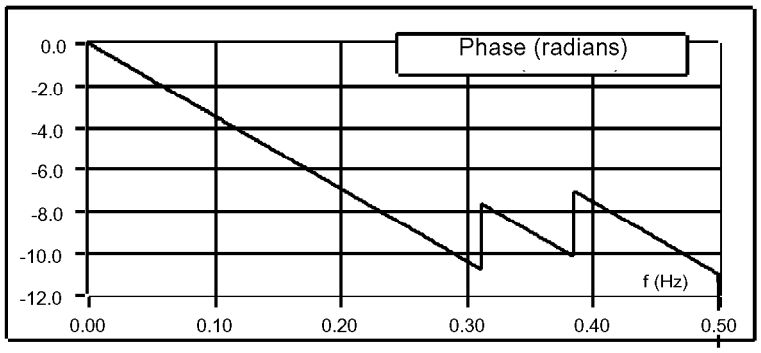
- They can achieve linear phase because of filter coefficient symmetry in the realization.
- They are always stable.
- You can perform the filtering function using the convolution and, as such, generally associate a delay with the output sequence

  $$\text{delay} = \frac{n-1}{2},$$

  where n is the number of FIR filter coefficients.

The following graphs plot a typical magnitude and phase response of FIR filters versus normalized frequency.





The discontinuities in the phase response arise from the discontinuities introduced when you compute the magnitude response using the absolute value. Notice that the discontinuities in phase are on the order of *pi*. The phase, however, is clearly linear. See Appendix D, *References*, for material that can give you more information on this topic.

You design FIR filters by approximating a specified, desired frequency response of a discrete-time system. The most common techniques approximate the desired magnitude response while maintaining a linear-phase response.

Designing FIR Filters by Windowing

The simplest method for designing linear-phase FIR filters is the *window design* method. To design a FIR filter by windowing, you start with an ideal frequency response, calculate its impulse response, and then truncate the impulse response to produce a finite number of coefficients. To meet the linear-phase constraint, by maintain symmetry about the center point of the coefficients. The truncation of the ideal impulse response results in the effect known as the Gibbs phenomenon – oscillatory behavior near abrupt transitions (cutoff frequencies) in the FIR filter frequency response.

You can reduce the effects of the Gibbs phenomenon by smoothing the truncation of the ideal impulse response using a smoothing window function. By tapering the FIR coefficients at each end, you can diminish the height of the side lobes in the frequency response. The disadvantage to this method, however, is that the main lobe widens, resulting in a wider transition region at the cutoff frequencies. The selection of a window function, then, is similar to the choice between Chebyshev and

Butterworth IIR filters in that it is a trade-off between side lobe levels near the cutoff frequencies and width of the transition region.

Designing FIR filters by windowing is simple and computationally inexpensive. It is therefore the fastest way to design FIR filters. It is not necessarily, however, the best FIR filter design method.

Designing Optimum FIR Filters using the Parks-McClellan Algorithm

The Parks-McClellan algorithm offers an optimum FIR filter design technique that attempts to design the best filter possible for a given number of coefficients. Such a design reduces the adverse effects at the cutoff frequencies. It also offers more control over the approximation errors in different frequency bands—control that is not possible with the window method.

Using the Parks-McClellan algorithm to design FIR filters is computationally expensive. This method, however, produces optimum FIR filters by applying time-consuming iterative techniques.

Designing Narrowband FIR Filters

When you use conventional techniques to design FIR filters with especially narrow bandwidths, the resulting filter lengths may be very long. FIR filters with long filter lengths often require lengthy design and implementation times, and are more susceptible to numerical inaccuracy. In some cases, conventional filter design techniques, such as the Parks-McClellan algorithm, may fail the design altogether.

You can use a very efficient algorithm, called the Interpolated Finite Impulse Response (IFIR) filter design technique, to design narrowband FIR filters. Using this technique produces narrowband filters that require far fewer coefficients (and therefore fewer computations) than those filters designed by the direct application of the Parks-McClellan algorithm. LabVIEW also uses this technique to produce wideband, lowpass (cutoff frequency near Nyquist) and highpass filters (cutoff frequency near zero). For more information about IFIR filter design, see *Multirate Systems and Filter Banks* by P.P. Vaidyanathan, or the paper on interpolated finite impulse response filters by Neuvo, et al., listed in Appendix D, *References*, of this manual.

Windowed FIR Filters

You use the **filter type** parameter of the FIR VIs to select the type of windowed FIR filter you want: lowpass, highpass, bandpass, or bandstop. The following list gives the two related FIR VIs:

- FIR Windowed Coefficients—Generates the windowed (or unwindowed) coefficients.
- FIR Windowed Filters—Filters the input using windowed (or unwindowed) coefficients.

Optimum FIR Filters

You can use the Parks-McClellan algorithm to design optimum, linear-phase, FIR filter coefficients in the sense that the resulting filter optimally matches the filter specifications for a given number of coefficients. The Parks-McClellan VI takes as input an array of band descriptions, each containing information describing the response you want for the given band. The VI outputs the FIR coefficients along with computed ripple, which is a measure of the deviation of the resulting filter from the ideal filter specifications.

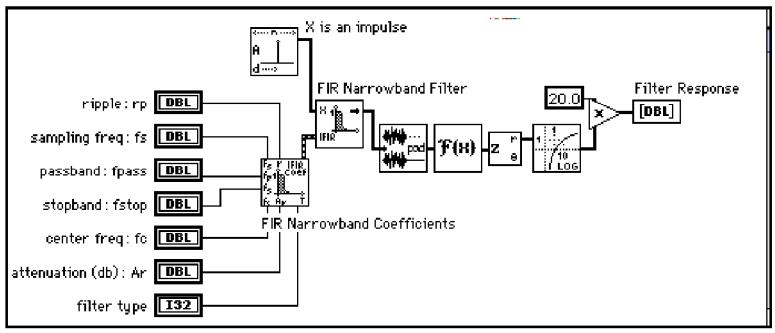
Four VIs use the Parks-McClellan VI to implement filters whose stopband and passband ripple level are equal: Equiripple LowPass, Equiripple HighPass, Equiripple BandPass, and Equiripple BandStop.

FIR Narrowband Filters

You can design narrowband FIR filters using the FIR Narrowband Coefficients VI, and then implement the filtering using the FIR Narrowband Filter VI. The design and implementation are separate operations because many narrowband filters require lengthy design times, while the actual filtering process is very fast and efficient. Keep this in mind when creating your narrowband filtering diagrams.

The parameters required for narrowband filter specification are filter type, sampling rate, passband and stopband frequencies, passband ripple (linear scale), and stopband attenuation (decibels). For bandpass and bandstop filters, passband and stopband frequencies refer to bandwidths, and you must specify an additional center frequency parameter. You can also design wideband lowpass filters (cutoff frequency near Nyquist) and wideband highpass filters (cutoff frequency near zero) using the narrowband filter VIs.

The following illustration shows how to use the FIR Narrowband Coefficients VI and the FIR Narrowband Filter VI to estimate the response of a narrowband filter to an impulse.



Nonlinear Filters

Smoothing windows, IIR filters, and FIR filters are linear because they satisfy the superposition and proportionality principles

$$L\{ax(t) + by(t)\} - aL\{x(t)\} + bL\{y(t)\},$$

where a and b are constants, $x(t)$ and $y(t)$ are signals, $L\{\bullet\}$ is a linear filtering operation, and their inputs and outputs are related via the convolution operation.

A nonlinear filter does not meet the preceding conditions and you cannot obtain its output signals via the convolution operation, because a set of coefficients cannot characterize the impulse response of the filter. Nonlinear filters provide specific filtering characteristics that are difficult to obtain using linear techniques. The median filter is a nonlinear filter that combines lowpass filter characteristics (to remove high-frequency noise) and high-frequency characteristics (to detect edges).

Filter VI Descriptions

The following Filter VIs are available.